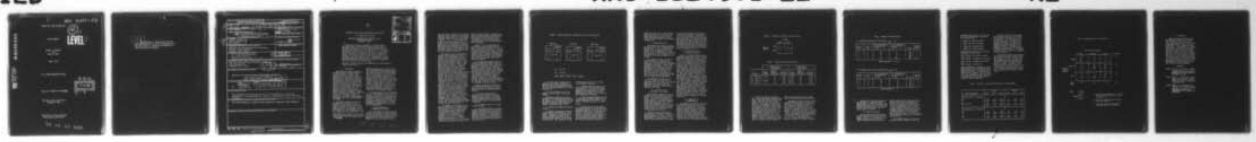
AD-A060 492 OHIO STATE UNIV COLUMBUS DEPT OF COMPUTER AND INFORM--ETC F/G 9/2
ESTIMATION OF RUN TIMES USING SIGNATURE TABLE ANALYSIS.(U)
JUN 78 S A MAMRAK, P D AMER DAAG29-77-G-0185

UNCLASSIFIED ARO-15249.1-EL NL

1 OF 1
AD A060 492



END
DATE FILMED
1-79
DDC

ARO 15249.1-EL

EARLY RUN TIME ESTIMATION

(12) NW

LEVEL II

FINAL REPORT

Sandra A. Mamrak
and
Paul D. Amer

June, 1978

U. S. ARMY RESEARCH OFFICE

Grant No. DAAG29-77-G-0185

The Ohio State University
Columbus, Ohio

Approved for Public Release
Distribution Unlimited

The findings of this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER (9) Final rept. 1 Aug 77 - 31 Aug 78	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Early Run Time Estimation • (6) Estimation of Run Times using Signature Table Analysis •	5. TYPE OF REPORT & PERIOD COVERED Final (August 1, 1977 - July 31, 1978)
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) Sandara A. Mamrak, Paul D. Amer	8. CONTRACT OR GRANT NUMBER(s) DAAG29-77-G0015 0185 new
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computer and Information Science Department, The Ohio State University, Columbus, Ohio 43210	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS U. S. Army Research Office, Post Office Box 12211, Research Triangle Park, NC 27709 (11)	12. REPORT DATE June 1978
	13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) The Ohio State University Research Foundation, 1314 Kinnear Road, Columbus, Ohio 43212 (12) 11 p.	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE NA

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

(15) DAAG29-77-G-0185

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

NA (18) ARO (19) 15249.1-EL

18. SUPPLEMENTARY NOTES

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Run time estimation, response time prediction, signature table analysis

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

407 121

/B

-A-

ESTIMATION OF RUN TIMES USING SIGNATURE TABLE ANALYSIS[1]

S. A. Mamrak and P. D. Amer

Department of Computer and Information Science
The Ohio State University
Columbus, OH 43210

Algorithms for managing jobstreams in a complex computer environment often rely on various estimates of job run times. Due to wide variability of run times from one execution of a job to another, point estimations of run times are fairly unreliable. An alternate approach to using point estimations is to use intervals which span the range of possible run time values. In an interval approach run times can be predicted with respect to membership in one of a limited set of run time intervals, with relatively high confidence. This paper presents a formal methodology for run time estimation based on an interval approach. The estimation is done using signature table analysis and is accompanied by a statement of statistical confidence in the results.

Key words: Interval estimation; point estimation; run time prediction; signature table analysis.

1. Introduction

Algorithms for managing jobstreams in a complex computer environment often rely on various estimates of job run times. Typical run times of interest include response time, processing time, turnaround time and so on. For example, scheduling algorithms which tend to minimize average job turnaround time based on the shortest-processing-time principle often rely on a prediction of what the job processing time will be. In systems which have a large degree of multiprogramming, run times for a particular job vary widely from one execution to another, depending upon the number and kinds of jobs that are simultaneously contending for resources. Prediction of run times, therefore, although fairly accurate "on the average," tends to be unreliable in any single instance because of the inherent complexity of the processing environment.

An alternate approach to using point estimations of run times, with their inevitably large variability and low confidence, is to use intervals which span the range of possible run time values. In an interval approach, run times are predicted with respect to projected membership in one of a limited set of run time intervals. The potential advantages of this technique are that in some environments prediction can be done based on very little knowledge about a job, and the confidence of predicting membership in the correct interval can be very high. The usefulness of this interval approach has long been recognized in the computer community and several ad hoc implementations exist. The classification of jobs in IBM's job preprocessor called HASP, for example, has been achieved in some installations by placing jobs in classes A, B, C and so on, based on user supplied estimates of resource requirements. Essentially, these classes represent predicted run time intervals for their respective members.

This paper presents a formal methodology for run time estimation based on an interval approach. The estimation is done using signature table analysis and is accompanied by a statement of statistical confidence in the results. It may be true that for very complex systems, subjective (or even random) estimation is the best method. This paper discusses the improvement possible on subjective "guesstimates."

2. General Background: Signature Table Analysis

Run time estimation for single computer systems is an important performance question which can be formulated in the following way: given a specified computer hardware and software configuration, and a workload which is composed of a series of jobs to be run on

[1]This research was supported by the U.S. Army Research Office under grant number DAAG29-77-G-0185

that system, what characteristics of the jobs can best be used to predict their respective run times. More specifically, let a computer system workload, W, consist of a series of n jobs, P_i, i=1, ..., n, and assume there exists a set of m descriptors $d_1, d_2, \ldots, d_m$ for each P_i which characterize that job's behavior. Then, the question of interest is which subset(s) of these descriptors can be best used to predict an additional or "key" descriptor, namely run time, and what is the particular function of the critical descriptors which yields this prediction.

The nature of the run time prediction problem and the motivation for developing certain kinds of methodologies for its solution can be illustrated by placing the problem in the context of a large, production-oriented computer system. In this case, a certain number of production jobs are being run on a regular basis -- daily, weekly, monthly, and so on. These production jobs often consist of several different programs (for example, payroll runs which include not only the relevant salary calculations, but also check-writing routines and summary report routines), and require a variety of system resources. Further, due to security and deadline constraints, they are often run on a dedicated system. The production jobs are completely specified and their characteristics with respect to development, maintenance and run-time behavior, $d_1, d_2, \ldots, d_m$, can be determined in most instances. Now if a new production job, P_{n+1}, is proposed for implementation on the existing system, the specification of its required turnaround time becomes a critical factor upon which to base the decision to allow or disallow it. Some subset of the projected behavior characteristics of P_{n+1} may be known, and resources may be available to investigate others, in order to estimate the job's turnaround time. The questions of which characteristics are most important in prediction, what form the predictor should take, and with what confidence the prediction can be made, must then be addressed.

The computer run time prediction problem can be formulated in terminology that makes the application of a pattern recognition technique called signature table analysis appear extremely appropriate. In essence, this technique deals with manipulating a set of data which possesses a finite number of discrete features, as well as a "key" feature. Analyses are performed on a "training sample" for which values of all the features, including the key feature, are known. Prediction of the key feature is explored, by means of the specification of a derived (combined) feature set which approximates the key feature on the training data. The derived feature set can then be applied to other sets of data for which the key feature must be predicted.

Typically, in a run time prediction environment, a training sample or set of data is collected which consists of a finite number of workload characteristics, like CPU, I/O and core resource requirements, along with the known turnaround time of already existing production jobs. Turnaround time prediction may then be conceptualized as the problem of identifying the significant "features" among the d_i which best describes a job's turnaround time "pattern".

The signature table method of pattern recognition suggested by Samuel [SAM67] for use in machine learning problems, and further developed by Page [PAG75] is a hierarchical approach for the recognition of patterns which are described in terms of many features. The method provides a means by which features are exhaustively analyzed in subsets, each of which provides a derived feature. The derived features are combined to result in higher derived features which depend in a nonlinear manner on all of the original features. An example of the tabular structure which may result from applying the method to four features is shown in Figure 1. (Figure 1 is discussed in more detail below.)

The major advantages of the signature table method over other prediction techniques, and those that render it especially applicable to the run time prediction problem are:

1) the quality of prediction is improved as more independent features or descriptors are used (this is in contrast to some techniques possessing the counter-intuitive property that for a finite-sized training sample there is an optimal number of features),

2) it provides a natural way to deal with missing data,

3) it allows the analyst to introduce personal knowledge and intuition about the system into the calculation process (this capability may greatly reduce the amount of computation required; it is comparable to the analyst's capability in the design of fractional factorial experiments to indicate which variable interactions are important and which are not),

Figure 1. Signature Tables for One Combination of Four Binary Features

Table D_{12}

d_1	d_2		Derived D_{12}
0	0		0
0	1	f_{12} →	1
1	0		1
1	1		1

Table D_{34}

d_3	d_4		Derived D_{34}
0	0		1
0	1	f_{34} →	0
1	0		1
1	1		1

Table D_{1234}

D_{12}	D_{34}		Derived D_{1234}
0	0		0
0	1	f_{1234} →	1
1	0		1
1	1		0

$$D_{12} = d_1 + d_2$$

$$D_{34} = d_3 + \bar{d}_4$$

$$D_{1234} = \bar{d}_1\bar{d}_2d_3 + \bar{d}_1\bar{d}_2\bar{d}_4 + d_1\bar{d}_3d_4 + d_2\bar{d}_3d_4$$

4) in many cases it provides better prediction than multiple regression, at less cost; this is true in part due to the use of an interval estimate approach rather than a point estimate approach as previously discussed, and

5) it is applicable to data in all formats: numeric, symbolic, ordinal and graphical.

The heuristics developed by Page to implement this technique have been specified for binary (i.e., two-valued) feature values, and therefore for the recognition of binary patterns. Essentially, the methodology requires the following steps: First,

1. Determine the appropriate predictor features.

2. Determine the appropriate cutpoint of each feature, using measures of minimum entropy (information loss) and maximum prediction. Cut-points are needed to discretize continuous features and to manipulate the allowed number of discrete values of each feature. Then iteratively, at each (derived) feature level, until a single derived feature is obtained,

3. Determine feature subsets or "signature types" upon which derived features are to be based. In Figure 1, for example, features d_1 and d_2 are combined to derive feature D_{12}, and features d_3 and d_4 are combined to derive feature D_{34}. But various other combinations are possible.

4. Define the derived feature resulting from the respective combined features, using an appropriate quantization method. This method is symbolized by the f_i's in Figure 1.

Finally,

5. Extract the relationship between the original features and the derived feature as Boolean expressions which describe, with some known probability, relations inherent in the data. This process is illustrated in Figure 1 for the derived feature D_{1234}. The potential usefulness of such an expression in run time prediction becomes apparent when one observes that only any three of the four feature values need be obtained to

calculate the D_{1234} value. Hence, the signature table method is a way of using the data to evolve switching functions which discriminate between members of various classes (or binary values of the key feature in Page's work).

3. A Sample Application

The signature table analysis approach was used to solve a run time prediction problem for the U. S. Army. A description of the application of the method to the Army data will serve to demonstrate its basic simplicity and its effectiveness in achieving the objective of interval estimation of run time with a relatively high degree of confidence.

The purpose of the Army study was to develop a predictor for the total turnaround (TA) time of a proposed application (production) job, based on a set of projected job resource requirements. Data for the development of the predictor consisted of 412 observations on currently running production jobs. A single observation was provided in the form of a 5-tuple, V = (CPU time, turnaround time, punch I/O, tape I/O, disk I/O). The data were divided into a training sample of 284 observations and a test sample of 128 observations.

The experiment was broken up into 4 steps: 1) division of the key feature (turnaround time) into intervals, 2) cutpoint specification for predictor features, 3) computation of derived features and 4) analysis of the results. These steps are discussed in turn below.

3.1. Division of the Key Features into Intervals

The key feature, turnaround time, was divided into three intervals: less than 10 minutes, 10-20 minutes and more than 20 minutes. These intervals appeared to be natural divisions in the data and were not chosen based on any statistical considerations of appropriateness. Also, they seemed to be reasonable intervals for use in the decision making process which would follow turnaround time prediction--namely, whether or not to allow the proposed job to be developed and supported.

The three intervals were defined by two cutpoints, 600 seconds (10 minutes) and 1200 seconds (20 minutes). For experimental purposes each of these cutpoints was investigated in a separate stage. First, boolean functions were derived to estimate if a job's run time would be less than or greater than 600 seconds. Then another set of functions was derived to estimate if a job's run time would be less than or greater than 1200 seconds. The functions which predicted with the highest accuracy from each set, based on the training sample data, were then combined to derive a single function. As will be described later in the analysis of the results, this single function was used to predict into which of the three intervals a job's turnaround time fell.

3.2. Cutpoint Specification for Predictor Features

Each of the predictor features was discretized into two ranges for each of the two experimental stages, a "low" and a "high" range. All of the predictor features were positively correlated with the key feature in that a low predictor feature value 'predicted' a low turnaround time and a high predictor feature value 'predicted' a high turnaround time. Given a particular key feature cutpoint, the predictor feature cutpoints were chosen so as to minimize the total number of incorrect key feature predictions. Cutpoints were determined using the Statistical Package for the Social Sciences (SPSS) [NIE75]. Basically, frequency tables of the form shown in Figure 2 were computed for different possible predictor feature cutpoints. The value for which (b+c) was minimized was selected as the cutpoint value. Table 1 contains the cutpoints which were computed for the two stages of the experiment. Also tabulated are the number and percentage of the 284 training sample observations which were incorrectly predicted using each cutpoint. Note that even the best cutpoint value in certain cases resulted in a large percentage of incorrect key feature predictions. This is due to a predictor feature's inability to single-handedly forecast the pattern of job turnaround time.

3.3. Computation of Derived Features

The computation of derived features has been described and analyzed in [PAG75]. A description of the steps followed in this study will be provided here. In general, predictor features are combined to produce second level derived features. These in turn are combined to produce higher level features. The process terminates when enough of the original predictor features have been used to produce higher level features which can predict the key feature's interval value with a high degree of accuracy.

Figure 2. Derivation of Predictor Feature Cutpoints

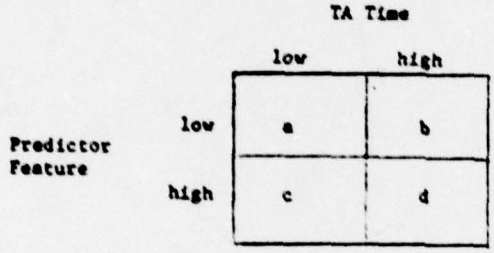


Table 1. Predictor Feature Cutpoint Values

Feature		TA Time Cutpoint: 600 Seconds			TA Time Cutpoint: 1200 Seconds		
		Feature Cutpoint	Number of Incorrect Predictions	% Incorrectly Predicted	Feature Cutpoint	Number of Incorrect Predictions	% Incorrectly Predicted
V1	CPU Time	60.0	25	8.8%	160.0	44	15.5%
V3	Punch I/O	1.0	123	43.3%	20.0	108	38.0%
V4	Tape I/O	400.0	53	18.7%	3110.0	57	20.1%
V5	Disk I/O	1.0	28	9.9%	1600.0	128	45.1%

Predictor features were combined in pairs. Since each feature had been divided into a low and high range by a feature cutpoint, there were four possible combinations: low:low, low:high, high:low and high:high. For purposes of computational ease, low was represented by 0 and high by 1. Once again using SPSS, frequency tables were computed to determine how many high and low key feature values existed in the training sample for each combination. For each combination the proportion of high values of the key feature, p_{high}, was compared to the proportion of high values of the key feature in the entire training sample. If the first proportion was larger, then it was judged that that combination predicted a high key feature value; otherwise, a low key feature value was predicted.

Two examples of derived features, one for the turnaround time cutpoint of 600 seconds and one for the turnaround time cutpoint of 1200 seconds, are provided in Table 2. It can be seen that a derived feature can be expressed as a boolean function or combination of the two features from which it was derived. In Table 2a, both Tape I/O and Disk I/O had to be 1 (high) for the derived feature to be 1. Consequently, the derived feature is equivalent to the boolean expression Tape I/O $\wedge$ Disk I/O, or more conveniently, V4 $\wedge$ V5. Likewise the boolean expression derived in Table 2b is CPU time, or simply V1.

Table 2. Examples of Derived Features

a. Turnaround time cutpoint is 600 seconds, $P_{high} = .845$

Tape I/O V4	Disk I/O V5	No. of observations with low TA time ($\leq$ 600)	No. of observations with high TA time (> 600)	P_{high}	Derived Feature
0	0	10	1	.091	0
0	1	18	36	.666	0
1	0	3	0	.000	0
1	1	13	203	.898	1

Boolean Expression: $V4 \wedge V5$

b. Turnaround time cutpoint is 1200 seconds, $P_{high} = .447$

CPU Time V1	Disk I/O V5	No. of observations with low TA time ($\leq$ 1200)	No. of observations with high TA time (> 1200)	P_{high}	Derived Feature
0	0	81	14	.147	0
0	1	54	8	.129	0
1	0	4	34	.895	1
1	1	18	71	.798	1

Boolean Expression: V1

The process for combining features was then repeated, this time combining the derived features. Eventaully, several final boolean expressions for both turnaround time cutpoints were determined, all of which were derived from at least three of the four original predictor features.

3.4. Analysis of Results

The final boolean expressions derived for each turnaround time cutpoint are presented in Table 3. For each expression, the number and percentage of correct and incorrect predictions have been tabulated. Based on the accuracy of prediction for the training sample, it was concluded that the variable V1 (CPU time) was the best predictor of turnaround time for both cutpoints. It should be remembered that the variable V1 used to predict below-above 10 minutes is slightly different from the variable V1 used to predict below-above 20 minutes inasmuch as different predictor feature cutpoints were calculated for each. The labels $V1_{600}$ and $V1_{1200}$ are employed below to differentiate between the two.

The variables $V1_{600}$ and $V1_{1200}$ were combined to derive a predictor of all three

turnaround time intervals. This predictor is a set of boolean expressions based on four variable values:

1. $V1_{600}$ → CPU time ≥ 60 seconds
2. $\overline{V1}_{600}$ → CPU time < 60 seconds
3. $V1_{1200}$ → CPU time ≥ 160 seconds
4. $\overline{V1}_{1200}$ → CPU time < 160 seconds

These variables were combined to form the turnaround time predictions:

$\overline{V1}_{600} \wedge \overline{V1}_{1200}$ → TA less than 10 minutes

$V1_{600} \wedge \overline{V1}_{1200}$ → TA between 10 and 20 mins.

$V1_{600} \wedge V1_{1200}$ → TA greater than 20 mins.

$\overline{V1}_{600} \wedge V1_{1200}$ → no prediction

The last combination is contradictory since $\overline{V1}_{600}$ implies CPU time less than 60 seconds and $V1_{1200}$ implies CPU time greater than or equal to 160 seconds. This combination was defined to be an automatic incorrect prediction. (As it turned out, none of the test or training sample data had this combination, an indication of the consistency of the separately derived expressions.)

Finally the accuracy of the predictor was estimated using the set of test data. Since turnaround times were available for the test data, it was possible to get an estimate of $p_{accuracy}$, the proportion of accurate predictions using the predictor. A summary of the actual turnaround times versus the predicted turnaround times is presented in Table 4. The left-to-right diagonal cells represent correct prediction since the predicted interval is the same as the actual interval. Other cells represent incorrect predictions.

Of the 128 test values, 101 observations were correctly predicted, thereby providing an estimate of the overall predictor accuracy, $p_{accuracy}$, of .789. An approximate 95% confidence interval for $p_{accuracy}$ was calculated, using a normal approximation, to be (.718, .860). In almost all cases (98.4% of the time), the prediction was either correct or within one interval. That is, seldom did the predictor predict less than 10 minutes when the actual turnaround time was greater than 20 minutes, and vice versa.

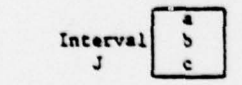
Table 3. Accuracy of Final Boolean Expressions on Training Sample

Boolean Expression	Correct			Incorrect		
	High	Low	Total	High	Low	Total
TA Time Cutpoint: 10 minutes						
V1	233 97.1%	28 63.6%	261 91.9%	7 2.9%	16 36.4%	23 8.1%
V1 ∧ V4 ∧ V5	201 83.7%	32 72.7%	233 82.0%	39 16.3%	12 17.3%	51 18.0%
TA Time Cutpoint: 20 minutes						
V1	105 82.6%	135 86.0%	240 84.5%	22 17.4%	22 14.0%	44 15.5%
V4	91 71.7%	136 86.6%	227 79.9%	36 28.3%	21 13.4%	57 20.1%

Table 4. Estimated Accuracy of the Predictor

Predicted TA Time (seconds)	Actual TA Time (seconds) 0-600	600-1200	1200 +	Row Total
0-600	13 81.3 86.7	2 12.5 4.7	1 6.3 1.4	16
600-1200	1 2.0 6.7	34 68.0 79.1	15 30.0 21.4	50
1200 +	1 1.6 6.7	7 11.3 16.3	54 87.1 77.1	62
Column Total	15	43	70	128

Legend



a: No. of TA values predicted to fall into interval J which had actual TA values in interval I

b: % of all J interval predictions which fell into interval I

c: % of actual interval I values which were predicted to be in interval J

4. Conclusions

Due to the large variability in job run times from one execution to another, point estimates of run times are unreliable. Interval estimation of run times is a reasonable approach to obtaining run time predictions in which a higher confidence can be placed. The application of signature table analysis to the prediction of turnaround time in one particular environment has yielded a predictor that was simple to develop, is simple to use, and is accurate about 80% of the time in predicting membership in one of three turnaround time classes. Although this interval approach technique will not provide sufficient predictive power for all applications, it is appropriate for some application objectives and should be considered as a desirable alternative to less statistically sound approaches.

References

[NIE75] Nie, N., C. H. Hull, J. G. Jenkins, K. Steinbrenner, and D. H. Bent, Statistical Package for the Social Sciences, 2nd edition, McGraw-Hill, Inc., New York, 1975.

[PAG75[Page, C. V., "Heuristics for Signature Table Analysis as a Pattern Recognition Technique," Computer Science Department, Michigan State University, 1975.

[SAM67] Samuel, A. L., "Some Studies in Machine Learning Using the Game of Checkers -- II -- Recent Progress," IBM Journal of Research and Development, Vol. 6, November 1967, pp. 601-617.